

CLAIMS

- 1 1. A method in a computer system for processing of a thread
2 waiting for access to a memory location, the method comprising:
3 when access by a thread to the memory location is blocked,
4 enabling an exception to be raised when that memory location
5 is accessed; and
6 blocking execution of the thread; and
7 when an exception is raised as a result of access by another thread to
8 that memory location,
9 completing the access by that other thread to that memory
10 location; and
11 restarting execution of the blocked thread.
- 1 2. The method of claim 1 wherein when access by the thread to
2 the memory location is blocked,
3 saving the state of the thread; and
4 storing a reference to the thread in the memory location.
- 1 3. The method of claim 2 wherein the stored reference is a
2 reference to a data structure that identifies the blocked thread and the saved state.
- 1 4. The method of claim 3 wherein the data structure indicates the
2 value that was stored in the memory location before storing the reference.
- 1 5. The method of claim 1 wherein when access by the thread to
2 the memory location is blocked, storing a reference to the thread in the memory
3 location.

667270-1299550

1 6. The method of claim 1 wherein multiple thread are
2 simultaneously blocked on the memory locations.

1 7. The method of claim 6 including storing a reference to the
2 blocked threads in the memory location.

1 8. The method of claim 7 wherein the stored reference is a pointer
2 to a list identifying the blocked threads.

1 9. The method of claim 8 wherein the list is a linked list.

1 10. The method of claim 1 wherein when the exception is raised as
2 a result of accessing the memory location, determining whether the access by the
3 blocked thread to the memory location will not block after access by the other thread
4 to the memory location and unblocking the thread when the access will not block.

1 11. The method of claim 1 wherein when the exception is raised as
2 a result of another thread accessing the memory location, determining whether the
3 access by the other thread will block and, when it will block, blocking execution of
4 that other thread.

1 12. A method in a computer system for deferring calculation of a
2 value until the value is accessed, the method comprising:
3 enabling an exception to be raised when a memory location is
4 accessed; and
5 when an exception is raised as a result of access to that memory
6 location,

652220"4379560

7 calculating the value associated with the memory location; and
8 providing the calculated value as the result of the access to that
9 memory location.

1 13. The method of claim 12 wherein the providing of the calculated
2 value includes storing the calculated value in the memory location and performing
3 the access to the memory location to access the calculated value.

1 14. The method of claim 12 wherein the value is calculated by
2 invoking a routine.

1 15. The method of claim 14 including storing a reference to the
2 routine in the memory location before the exception is raised.

1 16. The method of claim 12 including storing a reference to code
2 for calculating the value into the memory location.

1 17. The method of claim 16 wherein the stored reference identifies
2 parameters for passing to the code.

1 18. A method in a computer system for detecting access to
2 uninitialized memory, the method comprising:

3 enabling an exception to be raised when a memory location is
4 accessed; and

5 when an exception is raised as a result of access to that memory
6 location,

7 when the access is a write access,

8 disabling the raising of the exception; and

00361671.022299

9 allowing the write access to the memory location; and
10 when the access is a read access, indicating that access to
11 uninitialized memory has occurred.

1 19. The method of claim 18 wherein when speculative loads are
2 enabled, the indicating includes setting a poison bit for a destination register of the
3 read access.

1 20. A method in a computer system for detecting access to
2 protected memory, the method comprising:
3 enabling an exception to be raised when a memory location is
4 accessed;
5 when accessing the memory location with a trap for the exception
6 enabled, detecting and preventing access to that memory location; and
7 when accessing the memory location with a trap for the exception
8 disabled, allowing access to that memory location.

1 21. The method of claim 20 wherein when speculative loads are
2 enabled and the access is a read, the indicating includes setting a poison bit for a
3 destination register of the read access.

1 22. A method in a computer system for accessing a collection of
2 data items, the method comprising:
3 when adding a data item to the collection,
4 fetching and adding to a write counter, the fetched write
5 counter pointing to a bucket within a bucket array;
6 reading from the bucket pointed to by the fetched write pointer
7 using a synchronization access mode of sync;

09361671.072799

8 storing the data item in association with the bucket pointed to
9 by the fetched write pointer;
10 writing to the bucket pointed to by the fetched write pointer using a
11 synchronization access mode of sync; and
12 fetching and adding to a lower bound to indicate the number of
13 data items added to the collection.

1 23. The method of claim 22 wherein the bucket pointed to by the
2 fetched write pointer contains a pointer to a linked list of data items.

1 24. The method of claim 22 wherein the fetched write pointer
2 modulo a number of buckets in the bucket array points to a bucket within the bucket
3 array.

1 25. The method of claim 22 wherein the adding adds one to the
2 write counter.

1 26. The method of claim 22 wherein the adding adds a size of a
2 bucket to the write counter.

1 27. The method of claim 22 including
2 when removing a data item from the collection,
3 fetching and adding to a read counter, the fetched read counter
4 pointing to a bucket within the bucket array;
5 reading from the bucket pointed to by the fetched read pointer
6 using a synchronization access mode of sync;
7 removing the data item from association with the bucket
8 pointed to by the fetched read pointer; and

9 writing to the bucket pointed to by the fetched write pointer
10 using a synchronization access mode of sync.

1 28. The method of claim 22 including
2 when removing a data item from the collection,
3 checking the lower bound to ensure that the collection contains
4 a data item, when it cannot be ensured that the collection contains a data item,
5 indicating that the collection may be empty; and
6 when it can be ensured that the collection contains a data item,
7 fetching and adding to a read counter, the fetched read
8 counter pointing to a bucket within the bucket array;
9 reading from the bucket pointed to by the fetched read
10 pointer using a synchronization access mode of sync;
11 removing the data item from association with the bucket
12 pointed to by the fetched read pointer; and
13 writing to the bucket pointed to by the fetched write
14 pointer using a synchronization access mode of sync.

1 29. The method of claim 28 wherein the checking includes fetching
2 and adding a negative number to the lower bound.

1 30. The method of claim 29 wherein the checking includes fetching
2 and adding a positive number to the lower bound when it cannot be ensured that the
3 collection contains an item.

1 31. A method in a computer system for accessing a collection of
2 data items, the method comprising:
3 when removing a data item from the collection,

552220"74379550

4 fetching and adding to a read counter, the fetched read counter
5 pointing to a bucket within the bucket array;
6 reading from the bucket pointed to by the fetched read pointer
7 using a synchronization access mode of sync;
8 removing the data item from association with the bucket
9 pointed to by the fetched read pointer; and
10 writing to the bucket pointed to by the fetched write pointer
11 using a synchronization access mode of sync.

1 32. The method of claim 31 including before fetching and adding to
2 the read counter checking a lower bound to ensure that the collection contains a data
3 item.

1 33. The method of claim 32 wherein it cannot be ensured that the
2 collection contains a data item, indicating that a data item cannot be removed.

1 34. The method of claim 31 wherein the bucket pointed to by the
2 fetched read pointer contains a pointer to a linked list of data items.

1 35. The method of claim 31 wherein the fetched read pointer
2 modulo a number of buckets in the bucket array points to a bucket within the bucket
3 array.

1 36. The method of claim 31 wherein the removing adds one to the
2 read counter.

1 37. The method of claim 31 wherein the adding adds a size of a
2 bucket to the read counter.

09361671-072799

1 38. A method in a computer system for accessing a buffer of data,
2 the method comprising:
3 defining a write pointer to point to a location within the buffer;
4 when adding data to the buffer,
5 fetching the write pointer;
6 adding an indication of a size of the data to the write pointer;
7 and
8 storing the data into the buffer starting at a location indicated
9 by the fetched write pointer; and
10 setting the synchronization access mode of the write pointer to be
11 either normal or sync to effect the behavior of adding data to the buffer.

1 39. The method of claim 38 wherein the fetching and adding
2 includes executing a fetch and add operation.

1 40. The method of claim 38 wherein when the synchronization
2 access mode of the write pointer is set to normal, the storing includes overwriting
3 data previously stored in the buffer and not yet read.

1 41. The method of claim 38 wherein when the synchronization
2 access mode of the write pointer is set to sync, the storing includes waiting to
3 overwrite data previously stored in the buffer until the data has been read.

1 42. The method of claim 38 wherein the setting of the
2 synchronization access mode of the write pointer is transparent to the adding of the
3 data to the buffer.

09361671.0299660

1 43. A method in a computer system for implementing a circular
2 buffer, the method comprising storing in forwarding words located past an end of
3 the buffer pointers to locations at the other end of the buffer, the pointers having
4 forwarding enabled so that when a forwarding word is accessed, the access is
5 directed to the pointed to word at the other end of the buffer.

1 44. The method of claim 43 wherein the buffer is pointed to by a
2 write pointer whose value modulo a size of the buffer indicates the starting position
3 for storing data in the buffer.

1 45. The method of claim 43 wherein the buffer is pointed to by a
2 read pointer whose value modulo a size of the buffer indicates the starting position
3 for reading data from the buffer.

1 46. A method in a computer system for detecting access to a
2 memory location adjacent to a data structure, the method comprising:
3 storing a pointer to an invalid memory location in the memory
4 location;
5 enabling forwarding for the memory location; and
6 when access to that invalid memory location raises an exception,
7 indicating that the memory location adjacent to the data structure has been accessed.

1 47. The method of claim 46 wherein when speculative loads are
2 enabled, the indicating includes setting a poison bit in a destination register when the
3 access is a load from that memory location.

09361671.02299

1 48. The method of claim 46 wherein the access does not disable the
2 forwarding.

1 49. A method in a computer system for observing access to a
2 memory location, the method comprising:
3 under control of an observed thread, accessing the memory location
4 with a synchronization access mode of normal; and
5 under control of an observing thread,
6 accessing the memory location with a synchronization access
7 mode of sync; and
8 when access to the memory location is successful, indicating
9 access to the memory location by the observed thread.

1 50. The method of claim 49 wherein the observing thread disables a
2 memory retry-limit trap and when access to the memory location is not successful,
3 re-accessing the memory location with a synchronization access mode of sync.

1 51. The method of claim 49 wherein when the access to the
2 memory location is not successful, the observing thread is blocked until the
3 observed thread stores a value in the memory location.

1 52. A method in a computer system of restricting access to
2 memory, the method comprising:
3 setting a memory location to indicate a trap should occur when the
4 memory location is accessed;
5 under control of a restricted portion of a computer program,

654220" T 2373560

6 setting a pointer to point to the memory location that indicates
7 that traps to the pointed to memory location are enabled; and
8 accessing the memory location using the set pointer so that a
9 trap occurs and access to the restricted memory location is detected; and
10 under control of an unrestricted portion of the computer program,
11 setting a pointer to point to the memory location that indicates
12 that traps to the pointed to memory location are disabled; and
13 accessing the memory location using the set pointer so that a
14 trap does not occur and access to the restricted memory location is allowed.

1 53. The method of claim 52 wherein a user program typically
2 accesses memory locations using pointers with traps enabled.

1 54. The method of claim 52 including setting all memory locations
2 of a data structure to indicate a trap should occur when the memory locations are
3 accessed.

4 55. A computer-readable medium containing instructions for
5 causing a computer system to perform the method of claim 52.

6 56. A computer system including components to perform the
7 method of claim 52.

09361671.072999